WISCONSIN Information Retrieval For Business and Industry
TECH SEARCH

WTS Number: 492126

492126

RUSH

Request Date: 10/12/07 4:17 PM
Conf Number: 154197
Requester: Tim Butler
KENYON & KENYON LLP
1 Broadway
New York, NY 10004-

RUSH

Company Phone:
Requester Phone: 212-908-6166
Fax: 212-908-6113
Requester Email: tbutler@kenyon.com
Send-To Email: tbutler@kenyon.com
Reference: 10191/3800/dedi01

Delivery: Email
Instructions:

Bertram et al., "The Safety Related Aspect of Cartronic," SAE Technical Paper Series, International Congress and Exposition, Do. No. 1999-01-0488, March 4, 1999

3rd
SAE Microfilm
1999-01-0488

Copyright Royalty: $_____

Of Pages: _____

Refer Off Campus

SAE TECHNICAL
PAPER SERIES

1999-01-0488

The Safety-Related Aspect of CARTRONIC

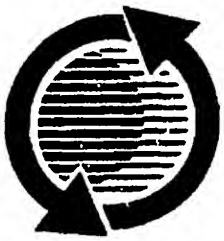
Torsten Bertram, Peter Dominke and Bernd Müller
Robert Bosch GmbH

Reprinted From: IV: Vehicle Navigation Systems and Advanced Controls
(SP-1428)

SAE The Engineering Society For Advancing Mobility Land Sea Air and Space®
INTERNATIONAL

International Congress and Exposition
Detroit, Michigan
March 1-4, 1999

400 Commonwealth Drive, Warrendale, PA 15096-0001 U.S.A. Tel: (724) 776-4841 Fax: (724) 776-5760



ISSN 0148-7191

Printed in USA 90-12038/PG

1999-01-0488

The Safety-Related Aspect of CARTRONIC.

Torsten Bertram
Peter Dominke
Bernd Müller
Robert Bosch GmbH

ABSTRACT

A networking of control systems poses high challenges - in particular for guaranteeing its safety, reliability, and acceptance of the whole integrated system by the car user. CARTRONIC is an open architecture for networking the control systems of a vehicle. The organization of a network has to be set up systematically and with foresight to achieve the advantages of going beyond the sum of the components and to avoid mutual disturbance. Thus the cooperation does not only require well-defined interfaces, but also coordination of the control strategies in the individual components. Additionally, there is an increasing need for ensuring that safety risks are effectively minimized, and for ensuring that no degradation in performance from either a safety or environmental point of view might take place. The paper is focused on the safety-related aspect of CARTRONIC, the safety analysis. The output of the safety analysis is a Safety Architecture. The Safety Architecture itself is based on the Function Architecture and adds from a safety point of view what is required to guarantee the safety of networking systems. The whole approach of the safety analysis is described in the form of a flow diagram.

1. INTRODUCTION

1.1 CARTRONIC

The automobile industry has reached a point where electronic systems which were so far essentially autonomous now grow together. This process is mainly driven by the demand for improved functionality, while at the same time limiting costs. Extended communication helps to make more intelligent use of, or even to simplify, existing systems. On the other hand, the networking of control systems poses high challenges - in particular for guaranteeing its safety, reliability, and acceptance of the whole integrated system by the car user. CARTRONIC is an open architecture for networking the control systems of a vehicle developed by Bosch. The function-related aspect of CARTRONIC was represented at the SAE'98 International Congress [Bertram et al. 1998]. The Function Architecture structures the logic of all known control tasks in the vehicle and is open for future extensions. The essence of CARTRONIC is to define structuring rules, modeling rules, and patterns for total, integrated control of vehicles.

The principles of the CARTRONIC Function Architecture are:

- Each system/component is made up of self-contained components with a minimal number of physical interfaces.
- Each system/component fulfills clearly defined tasks autonomously by obtaining information and initiating orders.
- Super-coordinated decision makers are used to coordinate systems/components.
- For example these coordinators are responsible for deriving one single information from competing requests.
- Orders are propagated hierarchically from initiator to actuator.
- The interfaces of each system/component are known to as many other systems/components as necessary and as few as possible.

The traditional example of communication in the area of active driving stability is the link between brake system and engine management for traction control. At the moment, adaptive cruise control pushes networking in the vehicle to a new level. In the long run, all longitudinal, lateral, and vertical vehicle control systems will integrate into overall vehicle guidance reaching out for safe accident-free driving.

The organization of a network has to be set up

systematically and with foresight to achieve the advantages of going beyond the sum of the components and to avoid mutual disturbance. Thus the cooperation not only requires well-defined interfaces, but also coordination of the control strategies in the individual components. The networking of all systems offers new driving capabilities and experiences.

Additionally, there is an increasing need for ensuring that the safety risk associated with the introduction and integration of new systems into the vehicle structure is effectively minimized, and for ensuring that no degradation of performance from either a safety or environmental point of view might take place.

1.2 HISTORY OF SAFETY ANALYSIS [WANG ET AL. 1997]

In the early era of system design, safety and reliability aspects received only limited consideration. The safety and reliability design was largely intuitive and based fundamentally on the designer's experience and skill. The statistically related techniques for safety and reliability analysis deliver a device for a more structured analysis. These methods gained popularity in the 1940s and 1950s. In the early 1960s the Failure mode and effects analysis (FMEA) was devised and safety and reliability analysis got a new driving force.

Awareness of safety concerns became essential to developers of high technology. Potential accidents, classified in terms of consequences and frequency of occurrence, were considered during the development process for the first time. It also became clear that integrated studies during the whole approach were needed to detect and reduce potential hazards of large and complex systems.

During the 1970s, FMEA was extended through potential accident scenarios. The scenarios covered system failures as well as operator errors during driving, testing, and servicing. Beginning in the late 1970s, the car industry adopted safety and reliability assessment techniques. Thus, applications were found across a range of activities and systems with different technological structures. Probabilistic safety, reliability, and availability criteria were increasingly used. Everywhere, safety criteria began to play a major role during initial stages of system design. As designers began to rely more on computers and simulation techniques, greater numbers of analysis techniques were incorporated into different approaches. The importance of failure and repair data collection programs was also realized alongside the adaptation of safety, reliability, availability, and maintainability assessment techniques during the 1980s.

Traditionally, safety analysis has been used primarily for verification purposes. Such an approach fails when designing complex systems with significant elements of novelty. For such systems, safety aspects need to be systematically integrated into the development process.

1.3 RELIABILITY AND SAFETY ANALYSIS

Despite their considerable overlap, reliability analysis and safety analysis are different concepts. They both refer to the study of system and process failures or operability. Reliability studies an item's characteristics, measuring the probability that the item will perform a required function under stated conditions for a stated period of time. This item could be a piece of equipment, component, subsystem, or system. If such analysis is extended to encompass the consequences of failure in terms of possible injury or death the study is referred to as safety analysis.

A safety analysis is devoted to preventing a system's risks. Risk is a measure of a hazard's occurrence and associated consequences. As used to assess risks associated with systems, safety analysis works to answer following questions:

- What can go wrong?
- What are the consequences?
- How often will they occur?
- What preventive measures must be taken to detect and reduce risks?

1.4 CARTRONIC SAFETY ARCHITECTURE AND FMEA

The FMEA [MIL1629A 1984] is a standardized method for the estimation of risks, for the recognition and rating of potential failures, and the identification of possible actions. It has been developed by the American aeronautics industry and is mostly used in the automobile industry. A FMEA is used to achieve, among others, the following corporate goals:

- Fail-safe and reliable systems.
- Reduced warranty costs and improved customer satisfaction.
- Reduced failure rate until series production is running to capacity.

Depending on the area of responsibilities there are three different types of FMEA, a system FMEA, a product FMEA and a process FMEA [VDA4/2 1996].

The system FMEA elaborates the correct cooperation of the physical systems and their components in the physical, technical environment. The analysis within the system FMEA is focused on the avoidance of failures which occur at the level of system definition and development of system concept. A prerequisite for a system FMEA is the existence of a system concept and a system definition. The system concept describes the partitioning of functions to physical components and

connections between them. A fundamental base for the system FMEA builds the system specification.

The product FMEA deals with the correct operation of physical components and products. This analysis is mainly influenced by a detailed performance specification and examines the abnormal behavior of each product independently. The goals of the product FMEA are the avoidance of malfunctions and influenced process failures which are caused by errors in construction. The analysis is based on knowledge and experience from earlier products during the design phase, and it is extended by experience from failures during the test phase. In order to get a detailed impression of potential failures, the failures are forced specifically.

The process FMEA elaborates the correct planning and realization of the design and implementation processes in order to avoid planning and manufacturing failures. The manufacturing process also includes a quality check. It is based on the product FMEA. If a process failure was detected during the product FMEA as a cause for a malfunction, then this failure is analyzed in more detail during the process FMEA and the malfunction is explored again.

The safety-related aspect of CARTRONIC follows an approach which is an abstraction of the system FMEA, and delivers a Safety Architecture from a functional and logical point of view. The output of the safety analysis according to CARTRONIC is thus valuable knowledge for the system FMEA. During the development of the Safety Architecture, the Function Architecture is analyzed under functional and logical circumstances. The Function Architecture is independent of a hardware topology so the safety analysis of components, communications, and their interactions deals with functional and logical failure scenarios.

The term component is used in the CARTRONIC Safety Architecture as well as in the Function Architecture in an abstract sense as an element of something bigger. During design, a component can be realized as software or hardware (physical component).

The Safety Architecture delivers functional and logical structures of components and communications which have to be added to the Function Architecture to guarantee safe running and operation. It includes the safe control of all consequences if a failure occurs.

1.5 OUTLINE

The paper is focused on the safety-related aspect of CARTRONIC, the safety analysis. The output of the safety analysis is a Safety Architecture. The Safety Architecture itself is based on the Function Architecture and adds from a safety point of view what is required to guarantee the safety of networking systems. The paper describes the approach which is used to analyze the Function Architecture and decide which safety strategies have to be applied.

The paper is divided into six main sections. Section 2 deals with possible failures in components or communications of the Function Architecture from a functional and logical point of view. It shows the consequences of these failures depending on the vehicle's behavior in a more abstract sense. Additionally, this section relates the three aforementioned aspects to each other in a schematic. Section 3 describes the relationship between possible failures in components or communications of the Function Architecture, the causes of failures from a functional and logical point of view, and the measures to detect and to deal with them. There is no rigid connection between an individual failure and an corresponding measure which has to be implemented. The choice of measures is influenced by safety, reliability, or economic factors. Section 4 illustrates the extension of the Function Architecture with functional and logical components and communications in order to realize the Safety Architecture in more detail. Section 5 outlines the approach in the form of a flow diagram to analyze a Function Architecture from a safety point of view and to add safety-related components and communications to detect and deal with failures. The flow diagram is defined in order to support designers during the development process. Section 6 gives a summary of the approach to analyze a Function Architecture and to develop a Safety Architecture. This section also shows some conclusions of a Safety Architecture for an Electronic Architecture.

2. CARTRONIC SAFETY ANALYSIS (CSA)

2.1 FUNDAMENTAL BASE

Incorporating safety into the early stages of the design process may in fact be the most effective way to reduce or eliminate potentially serious risks posed by large and complex systems. Safety analysis is a process of defining the need for safety; identifying, estimating, and evaluating risks; and conducting design reviews in order to reduce risks to an acceptable level. It aims to minimize injury and death. The process provides a systematic approach to the identification and control of high-risk areas in the system. When a large and complex engineering product is designed, the safety analysis identifies all possible failure conditions, assesses how frequently they may occur, and determines how serious consequences may be. This is achieved by first applying a functional and logical safety analysis according to CARTRONIC, followed by a system, product, and process FMEA.

In order to decide about the necessity and range of measures which have to be taken for failure detection and failure handling, the actual failure rate has to be compared with a maximum allowed frequency of failure events. The CSA compares actual values with the

allowed values in a more abstract way because it is independent of physical components. The approach is to classify the risks into different safety levels and to recommend a maximum allowed frequency of failure events for each safety level. In this case, the consequences are described in the form of a vehicle's behavior. The actual failure rates for the components and communications of a Function Architecture can be taken from experience of well known systems or from a FMEA. The process to estimate the real failure rates is not yet completely defined.

The CSA distinguishes between five safety levels. A fundamental base for the definition of safety levels for vehicle behavior is given by the International Standard for programmable electronic systems [IEC65A/179/CDV 1995, IEC65A/183/CDV 1995] and the German Standard for control of protecting devices [DIN V 19250 1989, DIN V 19251 1995] focusing on devices which are in a defined area, for example machine tools, handling machines, and roboters. A programmable electronic system is a computer based device, for example a "smart" sensor or actuator. On the other hand the German Standard considers control devices in safety systems. The main focus of both standards is on the endangerment of individuals.

2.2 SAFETY LEVELS

The system given in these standards is applied to the vehicle's situation. The definition of the safety levels of CSA has special emphasis on vehicle systems and deals with different scopes of application. For example, a vehicle can be driven on freeways, highways, or in urban areas. The different fields of application show that it is impossible to distinguish between the number of endangered persons near the vehicle. Another point in which the vehicle's situation can be distinguished from the general situation is that in a vehicle the probability of at least one person being near the car in case of a failure can always assumed to be one. This is not true for general machines where the probability of a person being near to it at night for example can be significantly lower than during the day. This results in a definition for vehicle behavior which is dependent on the endangering and injury level, and which is independent of the amount of people involved.

The classification of failure events and their consequences is determined initially by the level of injury (three levels), then by the possibility to control or to influence the situation (three levels), and by the frequency of occurrence (two levels) of a situation where the failure event causes the injury. The frequency of occurrence is formulated more abstractly by the circumstance, the failure event occurs in normal or special case.

The definition of the five safety levels of CSA are:

- *Safety level 4 (SL4):*
  1. **Injury level** —
     Immediate danger to body and life. (IL 1)
  2. **Possibility to control or to influence** —
     Neither controllable nor influenceable. (CI 1)
  3. Normal **case.**

- *Safety level 3 (SL3):*
  1. Immediate danger to body and life.
  2. Neither controllable nor influenceable.
  3. Special case.

  Or
  1. Immediate danger to body and life.
  2. Difficult to control or to influence. (CI2)
  3. Normal case.

- *Safety level 2 (SL2):*
  1. Immediate danger to body and life.
  2. Difficult to control or to influence.
  3. Special case.

  Or
  1. Immediate danger to body and life.
  2. Controllable or influenceable. (CI3)
  3. Normal case.

- *Safety level 1 (SL1):*
  1. Immediate danger to body and life.
  2. Controllable or able to influence.
  3. Special case.

  Or
  1. Slight injuries at most. (IL2)
  2. Opportunity to control or to influence irrelevant.
  3. Normal case.

- *Safety level 0 (SL0):*
  1. Slight injuries at most.
  2. Opportunity to control or to influence irrelevant.
  3. Special case.

  Or
  1. No immediate danger to body and life. (IL3)
  2. Opportunity to control or to influence irrelevant.
  3. Normal case.

The figure 1 shows the general schematic of the risk graph. The International Standard [IEC65A/179/CDV 1995] gives some hints as to who to select the allowed frequency of failure events and makes some suggestions for its values.

In addition to the International Standard the statistics of crash reports supplies valuable information about the range of real frequencies of failure events.

2.3 FAILURES IN THE FUNCTION ARCHITECTURE AND THEIR CONSEQUENCES

Figure 2 depicts the connections of the components and communications of the Function Architecture, the possible failures in components and communications from a functional and logical point of view, and the consequences of these failures depending on the vehicle's behavior in a more abstract sense. The consequences of failure events are strictly coupled with dangers for traffic participants. The dangers are assigned according to the above defined safety levels.

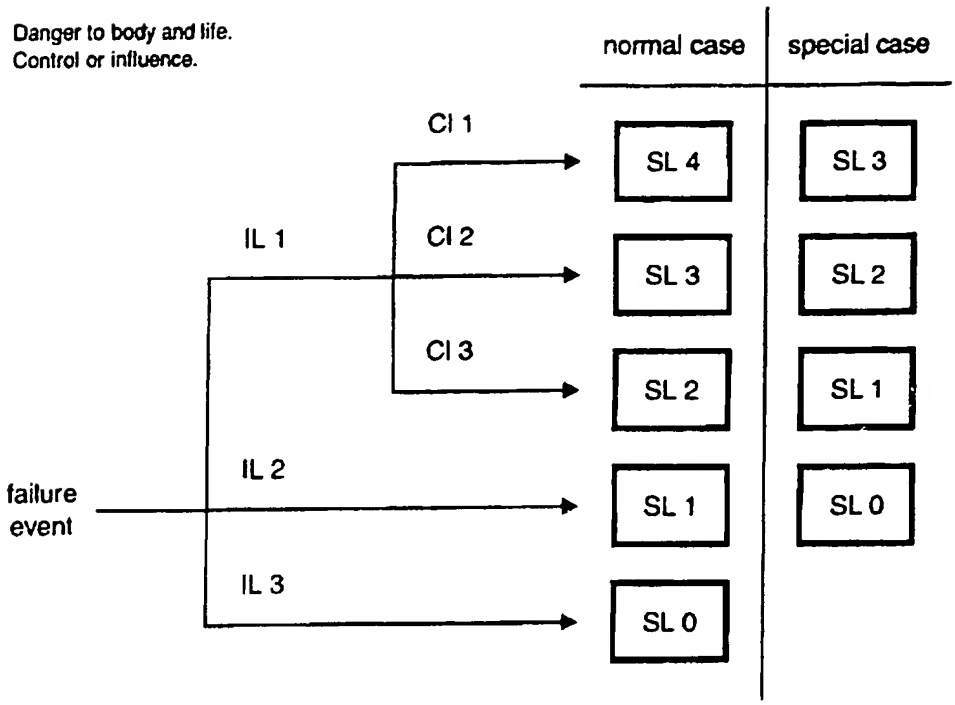


Figure 1: Risk graph.

The schematic represents a generalized matrix with three dimensions. One dimension describes the components and their communications of the Function Architecture under consideration in a required detailed scope. Another dimension defines the possible consequences in terms of the vehicle's behavior in case of failure.

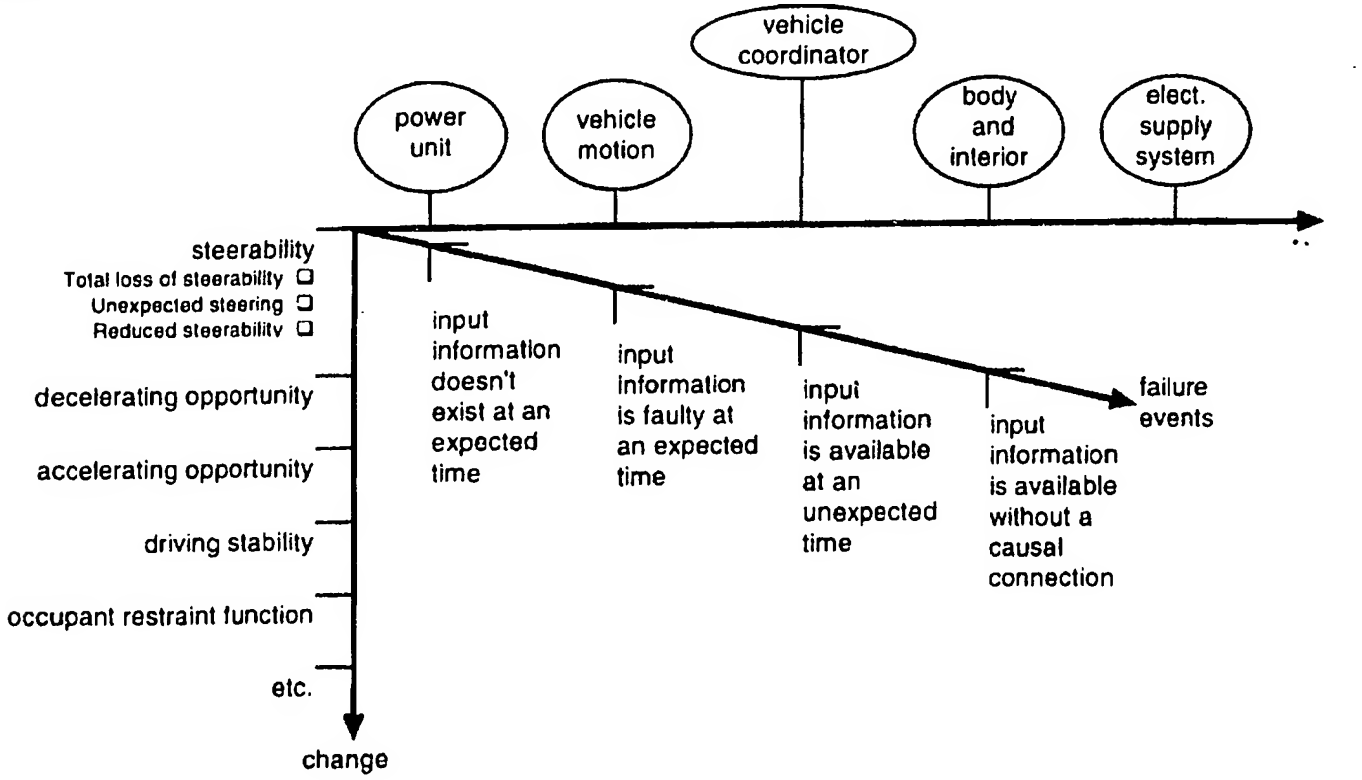


Figure 2: Schematic - Function Architecture, failure events, and consequences.

As mentioned above the consequences are classified into the five safety levels. The third dimension gives the possible failures in the Function Architecture. Each failure produces a new assessment level. This means for each failure the components and communications of the Function Architecture and the consequences of failure events define a plane in the matrix.

The level of abstraction of the Function Architecture which has to be considered in the schematic corresponds with the intention of the CSA. The level of abstraction can be adapted in any way. The process to adapt the level of abstraction can be realized in a recursive approach until these components and their communications are modeled for which concrete safety measures can be implemented and which from a safety critical point of view are mandatory. It is possible to determine for each component and communication on each abstraction level the failures and their consequences for the vehicle's behavior. With regard to the consequences of failures in the Function Architecture it is not necessary to distinguish between failures in components or communications. For example a failure occurs in a component and results in a faulty output value. This value is the input for another component. In the other case, if there is a failure in the communication, the consequence is also a faulty input value for the other component. In order to reduce the size of the matrix (schematic) for the failure analysis each communication is combined with its source component in the Function Architecture. The analysis of the consequences for the vehicle's behavior deals only with target components. The combination of communications and their source components supports the fact that the source of an information is obvious and the treatment of information in different components can vary. Regarding a realization of the Function Architecture the combination of component and communication does not mean that the physical reason for the failure is in the source component of the communication.

The generic failures on all abstraction layers in the Function Architecture are for example:

- An input information does not exist at an expected time.
- An input information is faulty at an expected time.
- An input information is available at an unexpected time.
- An input information is available without a causal connection.
- etc..

The information is a placeholder for the functional and logical kinds of communication, order, response, inquiry, and request.

The consequences of failure events are formulated in generic terms which describe the whole vehicle's behavior and can be labeled as follows:

- Change of steerability.
- Change of decelerating opportunity.
- Change of accelerating opportunity.
- Change of driving stability.
- Change of occupant restraint function.
- etc..

Depending on the detailed look each consequence can be more detailed structured, like for example:

- Total loss of steerability.
- Unexpected steering:
    - Controllable unexpected steering.
    - Uncontrollable unexpected steering.
- Reduced steerability.
    - Strong reduced steerability.
    - Slight reduced steerability.
- etc..

The structure of the schematic is organized in the way that it can be extended by additional attributes depending on the abstraction layer of the analyzed Function Architecture. The more detailed you look at the consequences the more sophisticated assessment and classification into the safety levels of the CSA get. The consequences on a more detailed look describes the worst that can happen. The worst that can happen is the worst case on a higher abstraction layer and it is expressed by the safety level.

When analyzing a failure mode f of a component A by analyzing (within the refinement of A) the internal failure modes that cause f, the worst consequence that can arise from one of the internal failure modes determines the safety level of component A with respect to the failure mode f.

Table 1 illustrates the relationships between components including their communications of a Function Architecture and consequences of failure events. The consequences are assessed depending on the safety levels of the CSA.

2.4 STATEMENTS FROM THE SCHEMATIC — FUNCTION ARCHITECTURE, FAILURES, CONSEQUENCES

The schematic supports the safety analysis during the development process. Safety-critical components and communications of a Function Architecture are detected and identified by the consequences in case of failure. The classification into safety levels represents the list of priorities to improve the Function Architecture from a safety point of view. For safety-critical components either measures to control or influence the consequences of a

failure event can be taken directly or the components must be more detailed in its inside to take the useful effective measures to handle the consequences of failure events. Beyond it a concrete assignment from failure to responsible component or communication is possible.

On a lower abstraction layer of the Function Architecture the consequences of a failure event can be assigned directly to a concrete component. On this kind of abstraction layer the functional and logical components represent sometimes concrete physical components.

consequences			safety level	Function Architecture								
				vehicle coordinator		power unit			vehicle motion		body and interior	electrical supply system
					engine	clutch	gearbox	propulsion /brake	steering	suspension		
steerability	total loss of steerability	...	4									
	unexpected steering	controllable ...	3									
		uncontrollable ...	4									
	reduced steerability	strong ...	3									
		slight ...	2									
decelerating opportunity	...	...	...									
		...	...									
accelerating opportunity	...	...	...									
		...	...									
driving stability	...	...	...									
		...	...									
occupant restraint function	...	...	...									
		...	...									
...			...									

Table 1: Relationships between components and consequences of failure events.

For example a component is from a functional and logical consideration an information provider and is realized as a sensor from a physical consideration.

The schematic delivers knowledge of redundant functions in the whole networking of systems. For example the braking system has the main task to decelerate the vehicle. If each wheel is decelerated individually then the whole vehicle can be steered.

The individual braking of wheels can be a consequence of a failure event or can be a controlled action in order to steer the vehicle in order to control the consequences of a failure event. So the braking system can provide a redundant steering system.

Besides the schematic provides some information for the correct partitioning of functional and logical components and communications to a net of ECUs, because the safety-critical components are identified.

2.5 EXAMPLE: RAW STRUCTURE OF PROPULSION AND BRAKE

Figure 3 depicts the raw Function architecture for propulsion and brake. The component "coordinator propulsion/ brake" is responsible for the coordination with regard to wheel torque. A positive wheel torque represents the acceleration case and on the other hand a negative torque stands for the deceleration case.

Possible failure events in the Function Architecture are:

- Component "coordinator propulsion/brake" and its communications.
  - Data transfer [M_{1P}, M_{2P}, M_{3P}, M_{4P}] is faulty. F1
  - Data transfer [M_{1B}, M_{2B}, M_{3B}, M_{4B}] is faulty. F2
  - Data transfer [$M_{P,MAX}$, $M_{P,MIN}$] is faulty. F3
  - etc.
- Component "propulsion".

- Data transfer [$a_{P,SOLL}$] is faulty. F4
- Realization [M_{1P}, M_{2P}, M_{3P}, M_{4P}] is faulty. F5
- etc.

- Component "brake".
  - Data transfer [$a_{B,SOLL}$] is faulty. F6
  - Realization [M_{1B}, M_{2B}, M_{3B}, M_{4B}] is faulty. F7
  - etc.

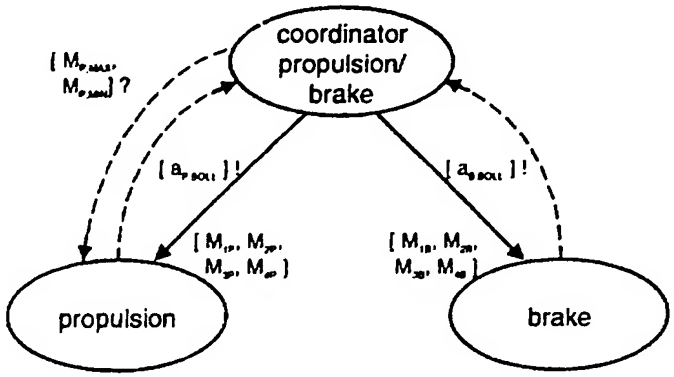


Figure 3: Function Architecture propulsion/brake.

Legend:

M_1	wheel torque at the front left.
M_2	wheel torque at the front right.
M_3	wheel torque at the back left.
M_4	wheel torque at the back right.
$a_{P,SOLL}$	desired vehicle's acceleration.
$a_{B,SOLL}$	desired vehicle's deceleration.
P	propulsion.
B	brake.
MAX	maximum value.
MIN	minimal value.

Table 2 points out the aforementioned failures and their consequences for the vehicle's behavior.

conse-quences			safety level	Function Architecture: coordinator propulsion/brake	Function Architecture: propulsion/brake propulsion	Function Architecture: brake
steerability	total loss of steer-ability	...	4	F2 front axle blocks		F6, F7 front axle blocks
	unex-pected steer-ing	control-lable ...	3	F2 one wheel brakes		F7 one wheel brakes
		uncon-trollable ...	4			
	re-duced steer-ability	strong ...	3	F2 front axle brakes		F6, F7 front axle brakes
		slight ...	2			
decelerating opportunity	total loss	...	4	F2		F6, F7
	unex-pected	strong ...	3	F2 front/rear axle brake		F6, F7
		slight ...	2	F1 powered axle brakes	F5 powered axle brakes	
	re-duced	...	2	F1, F2, F3 engine accelerates	F4, F5 engine accelerates	F6, F7
accelerating opportunity	total loss	...	3	F1, F2, F3 sum of torque = 0	F4, F5	F6, F7
	unex-pected	...	3-4	F1	F4, F5	
	re-duced	...	2	F1, F2, F3 brake torque > 0	F4, F5	F6, F7
driving stability	total loss	...	4	F1, F2 rear axle blocks	F4, F5 rear axle blocks	F7 real axle blocks
	re-duced	strong ...	3		F4, F5 rear axle spins	F6, F7 axles block
		slight ...	2	F1 real axle spins		
occupant restraint function	...	...	...			
		...	...			

Table 2: Safety analysis propulsion/brake.

3. FAILURES AND MEASURES

3.1 ANALYSIS OF CAUSES

The above analysis has identified the possible failures in the Function Architecture and their respective criticality. Usually the effectiveness of a measure in avoiding, detecting or dealing with the failure depends on the cause of it. Thus a more or less detailed analysis of the failure causes is necessary. As long as we deal with a Function Architecture the basic system where a failure can occur is shown in Figure 4.

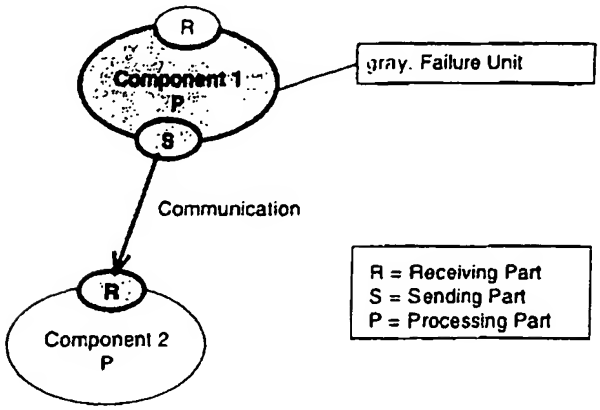


Figure 4: Basic system.

The gray shaded parts here give the failure-unit for the failures of the Function Architecture that have been discussed in section 2. Note that the communication link consists of the three parts S = sending unit, R = receiving unit and the channel itself. Given this picture the cause for the failure can either be in the communication part or in the component itself. As the component possibly can be refined and within the same structural elements are present, on the next refinement level the failure of the component can either be localized in a sub-component or in a communication link. In principle this procedure can be continued until we end up with components that cannot (reasonably) be refined within the Function Architecture, i.e. that are elementary in this context. This structured analysis is visualized in the Figure 5.

Although we do not want to emphasize the differences between failures, faults and errors [Laprie 1992], it is quite obvious that a failure of a sub-component leads to an error in the relevant component, thus is a fault of it.

Now when considering measures against the failures, many of them do not have to be implemented at the lowest level in the above hierarchy. For instance if we duplicate a component to provide redundancy the duplicated component can be introduced on every level. If duplication is an appropriate measure the refinement level on which it takes place must be chosen carefully

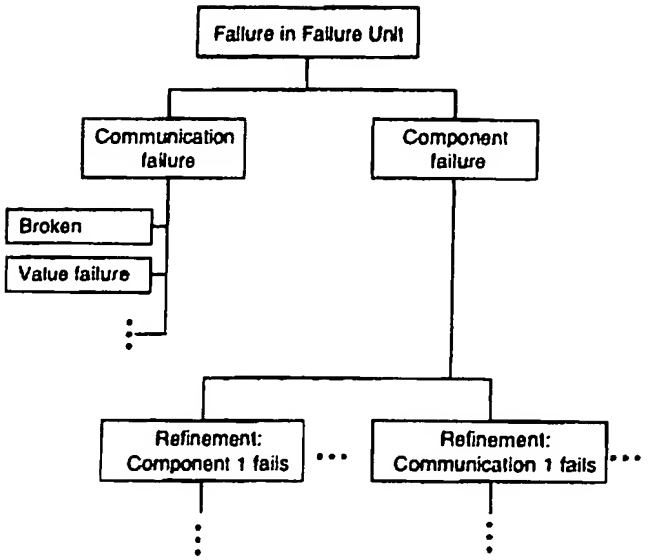


Figure 5: Refinement of components.

regarding the impact on cost, performance and fault coverage.

The possible (physical and non-physical) causes of a failure can be quite various. However, one of the advantages of the chosen abstraction is, that the effects can be classified. Thus we only have the following failure modes:

Communication failures

- Broken (no information transfer possible).
- Value failure (wrong value).
- Uncontrolled active (Communication link itself produces message. Produced information can be random for instance. It is not a (designed) system message).
- Timing failure (wrong time).
- Misdirection (message sent to wrong receiver).

Component failures

- Dead (component produces no output for relevant channel).
- Value failure (wrong value).
- Uncontrolled active (Component produces some more or less random output (not a designed system message) on the channel).
- Timing failure (wrong time).

Note that not all of these failure modes are disjoint. For instance a broken communication link can be seen as a limiting case for both timing and value failures and an uncontrolled active communication system usually

produces timing or value failures. However, the way how one failure mode can be interpreted as another one depends on the realization, and as the advantage of having disjoint classes is not significant here anyway, we use the above types of failure modes.

3.2 WHAT, WHO, WHEN

The reason why we look at failures is that we do not want them to happen or at least to produce an undesired effect. Thus we have to consider measures against these failures once they occur. When discussing measures there arise three aspects each of which can be characterized by a wh-question. The first is "What?" and deals with the measure itself. We consider this question in more detail in the lists below.

The second is "Who?" and asks for that part or component which is responsible for a certain action of the system. An example can be: Who is responsible for turning off this component? Typical answers are the following.

- Driver or human user. The system gives the driver an information and the driver is expected to react somehow, for instance by going to the next possible parking position.
- System itself. Here for instance the fail silence aspect is covered. If error detection takes place within a component and the component can shut down itself then the failure behavior towards the outside of the component is silent.
- Other system. Not the erroneous component but another component reacts, typically a component that is higher in the Function Architecture hierarchy. For communication failures usually the receiver reacts.
- Service.

The third question is "When?" and covers all time aspects:

- Fault-tolerance time. In most cases there is a time difference (in seconds, cycles or another unit) between the occurrence of a fault and the undesired effect produced by it. With respect to availability and robustness against transient faults any action against a fault can take place between detection of the fault and the end of the fault-tolerance time.
- Driving state. In some cases the usefulness of a measure depends on the vehicle state. For instance in the state "vehicle accelerates" the measure "reduction of engine moment" should be considered more carefully than in the state "vehicle stands".
- History. If the occurrence of faults is stored in memory then the time when a measure is initiated can depend on the fault-history of the relevant system.

Now we deal with the "What?" question. We generally group the measures into failure detection and failure handling measures. In the following discussion of measures one should always bear in mind that here the focus is on measures that can be integrated in the Function Architecture. This does not mean that measures that cannot be formulated within the Function Architecture because they depend on the realization should be neglected.

3.3 FAILURE DETECTION MEASURES

1. Acknowledgment.

The receiver of a message sends an acknowledgment sign indicating for instance reception, status of the message or even status of the receiver itself back to the source. As a measure this can be more than the CARTRONIC message "Response" [Bertram et. al. 1998]). Obviously this measure can detect a number of possible failures, in particular communication failures, but on the negative side when done explicitly besides increasing information traffic this introduces the possibility of back-pressure from receiver to sender [Kopetz 1997].

2. Redundancy.

Two or more comparable results are available and the comparison reveals if an error is present or not. This can be realized in different ways:

- Two or more algorithms (identical or diverse) are applied to the same input.
- This can be done on the same hardware or on different ECU's.
- The same algorithm can be executed twice on the same hardware to identify transient faults (double execution).
- The redundancy can as well be provided by a model or a second algorithm using different sensor values which is only able to give row values or, in the worst case, situation independent plausibility bounds. These are only used to check if the values produced by the „first" algorithm should be refused or not.

3. Reference checks.

A question with a known answer is asked. To give the answer the system must run (parts of) the code which is used when fulfilling the regular tasks. If the calculated answer does not coincide with the a priori known answer a failure is detected. The same principle holds when a default electrical input is given to a system, the response function of the correctly operating system is known, and the actual response function can be measured.

4. Observation of transmission channels.

Here typical examples are parity checks, CRC sums, or Hamming Codes. Another possibility is the integration of additional information on sender, receiver or message ID

into the message so that for instance the receiver can find out if the message has originally been sent to him. Such an information not necessarily has to be given explicitly.

5. Observation of physical properties.

A typical example is a temperature sensor or an IDDQ-test, where a high temperature (respectively high current) of the chip indicates that the results cannot be trusted anymore.

6. Logical/temporal observation of program execution.

Temporal observation can be done by a watch dog, or in a time-triggered system by the communication partners. Logical observation is for instance realized by software that ensures by use of counters that certain parts of a program are executed in the correct order.

7. Dynamic Communication.

To test a function that outputs a fixed value over an extended period of time, one can require, that the output is formulated in different ways. For example: every second value is to be sent inverted or with a negative sign.

3.4 FAILURE HANDLING MEASURES

1. Redundancy.

As for failure detection this is certainly one of the most powerful measures and the examples given above are valid here as well. However, to use redundancy for failure handling an additional information must be available, at least when looking at duplicated objects, for in the case of different outputs, there must be a criterion how to choose the correct one. This additional information can be realized in various ways:

- Failure detection. The information, which of the two outputs is faulty, is available. This holds in particular, when both duplicated components are fail silent.
- Strategy. In some cases the situation allows to use a fault-tolerant algorithm (for instance take first or take higher value). For more than duplicated redundancy you can even take an average over all non-extreme values.

2. Shut down of system part.

The complete part of the system, that is influenced by the error is turned off. This is only possible if a failure detection measure indicates which component(s) are faulty and if there is a system part that can be shut down without leaving the rest of the system in a state, where it is not possible to operate. In particular this means that all interfaces of this part to the rest of the system can operate with it being turned off. In principle this means switching the system state to a back-up state. A back-up state here is defined as a system state of the vehicle:

- Which is not dangerous.
- Which provides functionality to a reasonable extent.
- Which can be reached be reached by degradation of functionality in the vehicle.

If looked at this from this point of view there are two extreme versions of back-up states given in 3. and 4..

3. Complete shut down of electronics.

This can only be done if the basic vehicle functionality still can be provided. This certainly is not possible with a steer- or brake-by-wire system. A typical example is an ABS system which still leaves a manageable vehicle when completely turned off.

4. Safe parking state of vehicle.

Note that not every parking state is safe. Usually parking on a street or in any hostile environment is unsafe.

5. System stays in faulty state.

In some cases it is not possible to do anything about a fault or one does not know about it, so this is here for completeness.

6. Remove error.

This includes any kind of repair mechanisms. For transient faults a typical example is a reset of the ECU. Another more hardware oriented example is the burning free of electric contacts to remove the increase of a electrical contact resistance.

7. Change of strategy.

In case of failure the system can choose a strategy to handle the failure and to reach the goal given in the strategy with the means at disposition. These means are present in the Function Architecture, but they might be used in a different way in the non-failure case. To give an example (lying somewhere in the future): If the system detects the failure "complete loss of the braking system" and observes an obstacle being in the path of the vehicle it can change the strategy from "Driving according to driver's wishes" to „Avoiding an accident", by using the steering system to steer around the obstacle. Although the problems associated with such a measure certainly are serious (consider for instance validation) it should not be neglected in the future.

8. Additional elements.

This is a means to treat symptoms of faults before severe consequences occur. Consider for instance driver

information or an additional component that limits the effect of a failure. An example is a cooling ventilator that reduces the effects of a failure induced overheating of the computer.

9. Logical/temporal observation of program execution.

Although this being mainly a failure detection measure, in some cases where ignoring of a false output is sufficient to deal with it this measure can be used for failure handling. Note that it turned out that in most cases a time-triggered system has significant advantages concerning failure detection and handling.

3.5 DECISION FOR A MEASURE

Given the failure as analyzed above the first question always is if a measure is necessary. If answered positively one should look at the failure analysis and decide on which abstraction levels a measure can be implemented. As we cannot expect a unique answer we usually will have to iterate to find an optimum solution. A general rule of thumb (certainly not always true) is to integrate the measure as low as possible but as high as necessary. The reason for this rule is both cost optimization and to provide replaceability. Once we have decided on which level we want to introduce a measure the next question is which measure to take. The problem whether a failure handling measure is to be considered or a failure detection measure is sufficient might depend on the abstraction level or the application problem. In any case the first criterion is whether the measure taken into account is able to solve the failure problem or not. As we have a list of failure modes and a list of measures it is possible to construct a table indicating for each failure mode which measures can be used to deal with it. However, as most of this information depends on the situation and thus needs explanation we will not present this table here. In a given failure situation (which we have as result of the analysis above) it should be fairly easy to decide if a measure is able to deal with the problem or not. To decide which measure to take one has to consider performance of the measure (does it cover all aspects of the failure), effects on reliability and availability ("do not always use a complete shut down of the vehicle") as well as on maintainability and cost (see later for a discussion of the iteration loop).

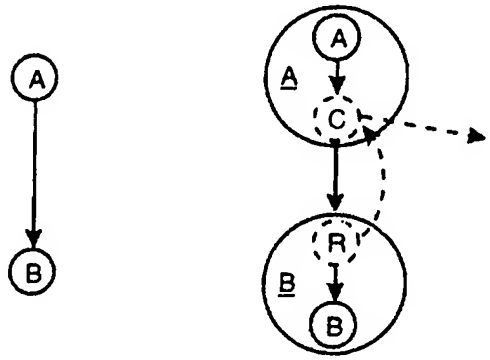
4. INTEGRATION OF MEASURES IN THE FUNCTION ARCHITECTURE

This section deals with the problem how to integrate the measures we have chosen according to the previous sections into the Function Architecture. As in complex systems safety considerations should be an integral part of the whole system the possibility of integration of safety measures into the Function Architecture is an important issue. In this context the following aspects have to be considered:

- The extensions introduced by the safety features thus can be governed by the same rules that hold for the Function Architecture.
- The safety measures can bring new components and new communication links into the Function Architecture.
- Already present components and communication links can be modified.
- The extension of a component due to a measure modifies only the component itself.
- The extension of a communication link modifies no other part of the Function Architecture except the link, the source and the target component of the link.
- New components that are exclusively used for failure detection but can detect failures in more than one component shall be treated like an information provider in the Function Architecture.
- There can be more than one way to integrate a measure.

We cannot show for every measure discussed above how to integrate it into the Function Architecture, but we give some illustrating examples. In the following pictures on the left hand side the situation in the Function Architecture without a safety measure is shown and on right hand side the situation with a safety measure. New components and communication links are dashed.

Acknowledgment

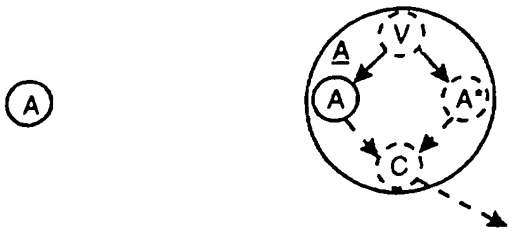


The components A and B are replaced by A and B. In B a new component R is responsible for receiving the message sent by A and sending the acknowledgment back. In A the new component C has to compare the acknowledgment message with the message originally sent. In case of an error, C has to send an error message. This error message can stay within A, but can as well have any other legal target within the Function Architecture.

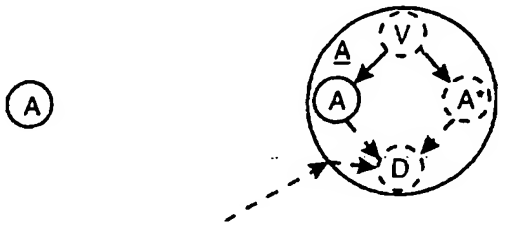
Failure Detection: Component Redundancy

The component A is replaced by A. Within A the distribution element V gives the task originally performed by A to both A and A'. These send their results to the comparator C which produces an output if both results

coincide or an error detection message to some legal target within the Function Architecture if not.

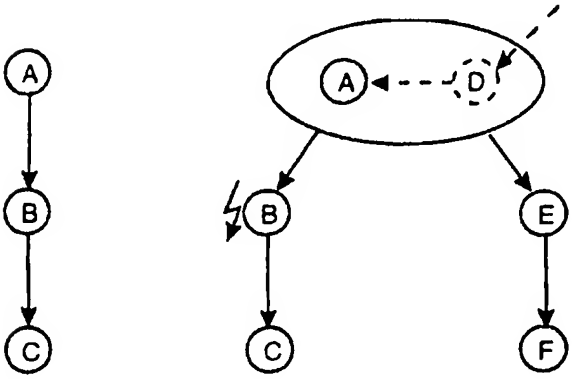


Failure Handling: Component Redundancy



The difference to the picture above is, that the comparator C is replaced by a decision element D which decides in case of a discrepancy between the outputs of A and A* what the output of the component A is. The decision element can use some kind of third information like a failure detection information or an internal strategy to do its task.

Change of Strategy



The normal flow of information is given on the left hand side (from A to B to C). In case B is faulty the decision element D can inform A, that a strategy change has to occur and that A should give E a task according to the new strategy. The information on which D decides must originate from some legal source within the Function Architecture.

5. FLOW DIAGRAM FOR CSA

5.1 CORRELATION BETWEEN FUNCTION AND SAFETY ARCHITECTURE

The development of a Function Architecture for a car function is influenced by the safety analysis of this function. The first step is the structuring and definition of components and communications. After the row Function Architecture has been developed the safety analysis starts to detect safety-critical elements of the Function Architecture. The approach to develop a Function Architecture and to analyze this architecture under a safety point of view is a simultaneous procedure. The functional and logical analysis which results in a Function Architecture starts a little earlier than the safety analysis. The simultaneous procedures function and safety analysis show that a clear separation between the two aspects is impossible. Besides each analysis influences the other one and end up with a recursive process. This means that the "optimal" Function Architecture only from a functional and logical point of view does not fit well with safety aspects. The vice versa statement is also true. The effort for the safety analysis depends on the safety-relevance of the analyzed car function and its influence on other functions. Depending on the safety-relevance it is possible that a Function Architecture is not enough detailed. In this case the safety analysis supports the function analysis to find a more detailed and precise Function Architecture. If a function is not safety-critical then a specification which defines minimum demands is enough for the safety analysis.

5.2 QUICK GUIDANCE FOR CSA

The goal of this part of paper is to define a quick guidance for designers to use the ideas which were described in more detail in sections 2, 3, and 4.

The flow diagram consists of eight steps (Figure 4):

1. Check the availability of Function Architecture for the analyzed car function and guarantee the understanding of the method CSA.

   The first step tries to guarantee that the Function Architecture which was developed and described under the constraints of the CARTRONIC concept for the functional and logical analysis was understood, and that the procedure for the CSA is well known. The procedure includes as well the definition of the allowed frequency of failure events.

2. Put together all possible consequences of failure events and classify the consequences into the safety levels.

   During the second step the designer has two alternatives to handle consequences. One alternative is to figure out the consequences in more detail, the other one is to consider the consequences in more detail. Which alternative to choose depends on the assignment of consequence to safety level. The analysis of consequences is a function-related approach and is independent of a realization. For this step information from a system FMEA for similar

systems provides valuable inputs.

3. Fill out schematic 1 (components, failure events, and consequences; see section 2) depending on the desired, detailed look for the Function Architecture.

   The third step is a time-consuming process, because depending on the abstraction layers there are a lot of components and communications which have to be analyzed in order to assess the failures and their consequences. During the analysis the designer gets a better feeling for the consequences and can check if the classes of consequences are detailed enough and if all possible consequences are considered. The question of abstraction which is in the focus of analysis should be answered carefully.

4. Identify the safety-critical failure events and components.

   The fourth step identifies the safety-critical components including their communications by the assignment of the consequence of a failure event to a safety level. The highest safety level means the highest danger to body and life. The main focus during the next steps is on the components of the highest safety level.

5. Formulate the causes for the failure events in form of a track until from a functional and logical point of view useful information is available.

   The main topic of the fifth step is searching for reasons why the failure occurs. The searching for reasons is a top down approach. The designer starts the safety analysis with the functional and logical components on a high abstraction layer. If there is a failure event detected then the designer has to look into the detailed structure of the component in order to find a more precise and local limited reason for the failure event. This procedure comes to a first end when the designer is able to make clear decisions about measures to detect the failure events and to control the consequences.

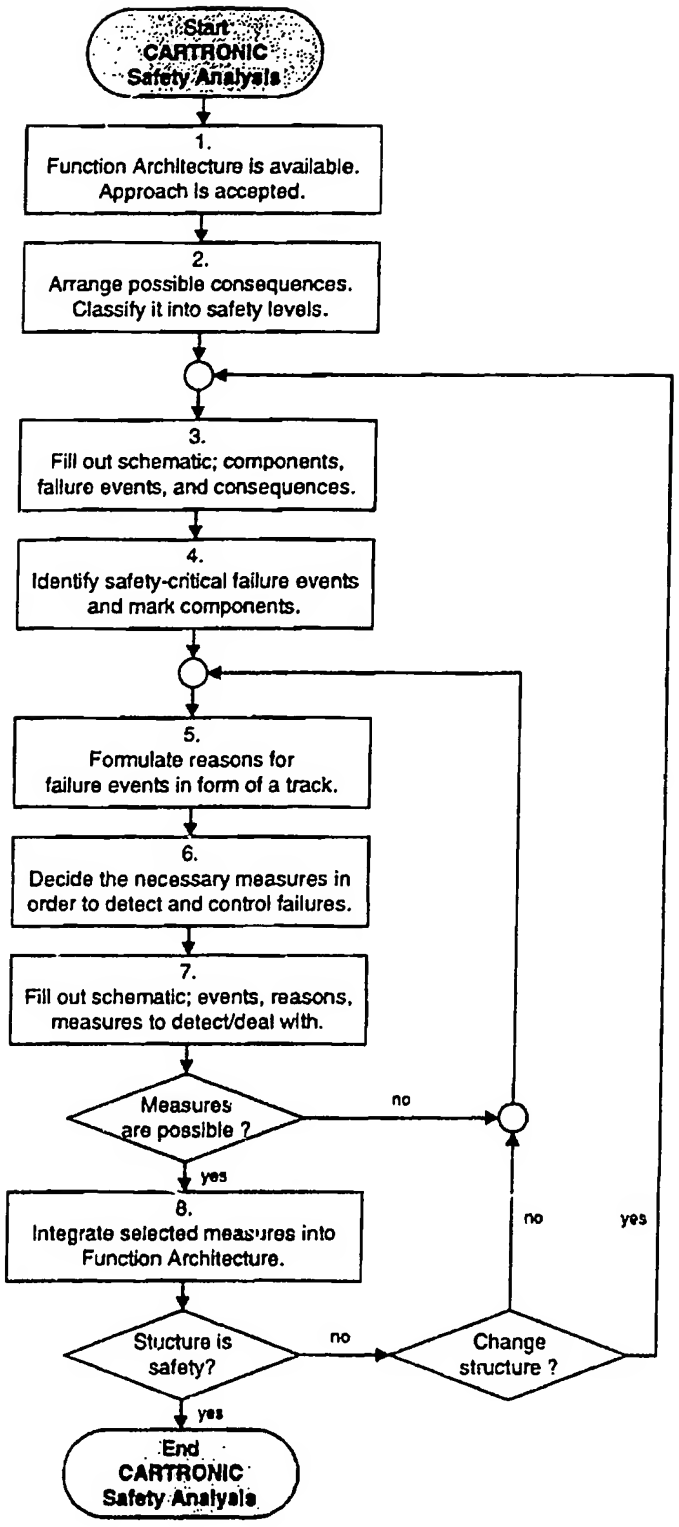


Figure 6: Flow Diagram for CSA.

6. Analyze measures and their ability to detect and to handle the failures and their consequences.

   The sixth step supports the elaboration of an expressive statement about the necessity and range of measures to deal with the consequences. The designer needs a realistic value of the real failure rates. Then he compares the real failure rates with the maximum allowed frequency of failure events and can determine the necessity and range of action. If it is not possible to get statistical information of similarly realized information then the designer has to make some assumptions depending on his experience and skill. If the designer has made faulty assumptions for the real failure rates then he is not able on the one

side to find the correct measures to limit the consequences. But on the other side the approach to find measures depending on the improved assumptions is capable of being enlarged and it fits in the flow diagram as before.

1. Define all measures for failure detection and handling (see section 3).

   The decision for a particular measure in order to detect a failure event or to handle the consequences is influenced by other criteria. There is not a strict assignment between a failure event and a measure. On the contrary there is a catalogue of measures for detecting and handling failures. The criteria for the decision are for example safety-relevance, costs, state of the art, acceptance, opportunity to realize, etc.. The results that are available at this point provide the starting point for a recursive approach, because they can be extended in each direction of the underlying matrix. After the decision for a measure the designer has to check the effectiveness and to figure out the reduced frequency of failure event for the analyzed structure. If the measures reduce the failure rate sufficiently, this part of the CSA is finished.

2. Realize selected measures.

   The eighth step integrates the measures into the existing Function Architecture. The result is a safety Function Architecture with some components and communications which are only safety-related. This means that the functional and safety aspects results in a common architecture and so the development processes for a Function and a Safety Architecture of car's functions are parallel and each other influencing processes.

   8.1. Integrate selected measures into the existing Function Architecture (see section 4).

   Or

   8.2. Change the existing Function Architecture to realize selected measures (see section 4).

The whole CSA is a recursive approach which starts with the Function Architecture and the arrangement of possible consequences. During the process some steps have to be taken multiple times in a short period within a recursive loop (Figure 4) in order to improve the safety of the analyzed Function Architecture. The recursive loops guarantee as well the effectiveness of measures to detect failure events and to handle their consequences. During the approach the designer stumbles over some questions which can be only answered making useful assumptions about the hardware-topology. Some measures to detect and deal with failures depend on their physical realization. This is reflected in the analysis of reasons for failure events.

6. SUMMARY AND CONCLUSIONS

In this paper it has been shown how to perform a safety analysis of a given Function Architecture. As a result it is possible to introduce measures into the Function Architecture in order to reduce or avoid the effects of failures discussed in the safety analysis. The integration of measures complies with the rules of the Function Architecture. Moreover, this procedure gives some hints on a possible distribution of Function Architecture components on different hardware. If for instance a back-up state you wish to use requires a shut down of a subset of components it can be an advantage to have those components on one piece of hardware, separated from the others. If such a mapping of Function Architecture components to hardware is possible, then it is fairly easy to realize the transition to the back-up state in the physical world. However, this certainly is an area to be discussed in more detail and thus is part of future work.

CONTACT

Robert Bosch GmbH
Dep.: FV/SLF
Dr.-Ing. Torsten Bertram
P. O. Box 30 02 40
D-70442 Stuttgart
Germany

Torsten.Bertram@pcm.bosch.de

REFERENCES

[Bertram et al. 1998] Bertram, T., R. Bitzer, R. Mayer and A. Volkart. 1998. CARTRONIC - An Open Architecture for Networking the Control Systems of an Automobile. 1998 SAE International Congress and Exposition. Detroit/U.S.A., 23.-26.02.1998. SAE 980200.

[DIN V 19250 1989] DIN V 19250. 1989. Grundlegende Sicherheitsbetrachtungen für MSR-Schutzeinrichtungen.

[DIN V 19251 1995] DIN V 19251. 1995. MSR-Schutzeinrichtungen, Anforderungen und Maßnahmen zur gesicherten Funktion.

[IEC65A/179/CDV 1995] International Electrotechnical Commission. 1995. Technical Committee No. 65: Industrial Process Measurement and Control. Sub-Committee No. 65A: System Aspects. General Requirements (Draft IEC 1508 - Functional safety: safety-related systems).

[IEC65A/183/CDV 1995] International Electrotechnical Commission. 1995. Technical Committee No. 65: Industrial Process Measurement and Control. Sub-Committee No. 65A: System Aspects. Guidelines

on the application of Part 1 (Draft IEC 1508 - Functional safety: safety-related systems).

[Kopetz 1997] Kopetz, H. 1997. Real Time Systems, Design Principles for Distributed Embedded Applications, Kluwer Academic Publishers Boston Dordecht London.

[Laprie 1992] Laprie, J.C. (ed.). 1992. Dependability: Basic Concepts and Terminology, Dependable Computing and Fault-Tolerant Systems Vol. 5, Springer Verlag Wien New York

[MIL1629A 1984] Military Standard: MIL1629A. 1984. Procedures for Performing a Failure Mode, Effects and Critically Analysis. Notice 2.

[VDA4/2 1996] VDA. 1996. Sicherung der Qualität vor Serieneinsatz. Band 4, Teil 2. System-FMEA. Verband der Automobilindustrie e. V.

[Wang et al. 1997] Wang, J. and T. Ruxton. 1997. Design for safety. Journal of American Society of Safety Engineers. January 1997, pp. 24 - 29.